Nuclear Mathematical and Computational Sciences: A Century in Review, A Century Anew
Gatlinburg, Tennessee, April 6-11, 2003, on CD-ROM, American Nuclear Society, LaGrange Park, IL (2003)

GIPGUI: A GRAPHICAL USER INTERFACE FOR THE GIP CROSS SECTION PREPARATION CODE

Douglas E. Peplow[*]
Oak Ridge National Laboratory
PO Box 2008 MS 6363, Oak Ridge, TN 37831
peplowde@ornl.gov

Yousry Y. Azmy
Department of Mechanical & Nuclear Engineering
Pennsylvania State University
229 Reber Building, University Park, PA 16802
yya3@psu.edu

ABSTRACT

Writing an input deck for the Group-organized cross-section Input Program (GIP) can be quite a challenge, especially for those who do not ordinarily work in discrete ordinates. Material compositions must be specified for each P_n component individually and only in terms of atomic densities, i.e. atoms/(b cm). In order to make the input of material information more logical and closer to physical reality, GipGui was written. This interactive utility allows the user to specify materials in a variety of ways, including molecular formula and weight fractions. The separate Legendre components are handled automatically, not as separate nuclides, so the user can focus on true materials. At present GipGui operates as an interface to standard GIP input files, but stops short of executing GIP, since the former activity presents the hardest challenge to many users of codes in the Discrete Ordinates of Oak Ridge (DOORS) package. GipGui is designed to be backwards compatible with minor restrictions, hence it reads and manipulates existing GIP input files, as well as allows the user to start from scratch.

Key Words: GIP, graphical user interface, discrete ordinates, cross sections

1. INTRODUCTION

GIP[1] is the standard cross-section manipulation code used in preparing cross-section data for the neutral particle transport codes comprising the Discrete Ordinates of Oak Ridge System (DOORS[2]) package, e.g. ANISN, DORT, and TORT.[3] All distributed versions of DOORS include the GIP code in addition to other peripheral codes that work in conjunction with the primary transport codes to address a broader class of problems encountered in applications.[4] The programming environment of the mid 1960's in which GIP was developed had tight constraints on the amount of memory that can be dedicated to execution of a given code. Hence much of the peculiarities of computer codes that were born in that environment appear unjustifiable in the present computational environment characterized by large physical and virtual memory size, and high processor speed. As an example, the treatment of the P_n components of the scattering cross

[*] Corresponding author

section for a specific nuclide as a set of distinct nuclides was an innovative solution designed to reduce memory requirement without introducing significant changes to the logic and structure of other related codes. As a result of such non-intuitive programming practices, the primitive nature of the programming languages in which many early neutronics codes were written, and poor user interfacing, GIP is viewed by many in the user community as a powerful and necessary but extremely difficult computational tool. GipGui is designed to remove this burden from the users by implementing an interactive intuitive interface between the user and standard GIP input files that is backwards compatible with a few restrictions. Hopefully, by using GipGui, the chances of introducing errors into the mixed cross sections will be reduced.

GipGui is written in Java in an effort to be platform independent. This is an alpha version that was written to see if there was a desire in the discrete ordinates community for such a tool. At the present time, GipGui only prepares the input deck, it will not run GIP. Future additions could include: ability to handle special instructions, such as those multiplying a cross section by a constant or operations dealing with just one Legendre component; ability to read an actual library instead of a library index file; ability to run GIP directly from the GipGui. There are also a number of data checks that GipGui does not handle that can be added to help the user create the actual GIP input file.

2. DETAILED DESCRIPTION

2.1. Coding Language

GipGui is written in Java 1.3 and distributed as a single JAR file (Java ARchive, similar to a unix tar file but executable).

2.2. Nature of Problem Solved

GipGui can create from scratch or modify existing GIP input deck text files. This interface creates the group-by-group instructions from the user's physical descriptions of materials.

2.3. Method of Solution

2.3.1. Library index file

GIP input decks refer to the materials that are listed in a library file, often a very large file. GipGui needs to know only about the structure of such a library, it does not need to access the actual cross-section data in the library. So, GipGui uses a library index file, which the user will have to create for each real library he uses.

The first line in the index file specifies the number of energy groups, the number of upscatter groups, the number of downscatter groups, the number of activation groups, total upscatter (0-no, 1-yes) and finally the total number of records in the library. Text comments may follow.

```
line 1 format: int int int int int int text_comments
```

Then, for each library material, specify in one line the record number of the P_0 component, the highest order component, a useful name, e.g. nuclide identifier, and any comments enclosed in quotes.

```
format: int int name "text_comments"
```

For example, an index file might look like this:

```
69 0 0 68 1 612 taken from last year's project
1    7 al27    "v94.10 standard wgt e61132"
9    5 am241   "v94.10 standard wgt e63954"
15   5 am242   "v91.94 standard wgt e62954"
601  5 y89     "v91.94 standard wgt e61392"
607  5 zr      "v91.94 standard wgt e62400"
```

Only materials that the user wishes to mix need to be listed, although line one must include the actual total number of materials in the full library whether they are used or not. Extra spaces don't hurt – they can be used for easier human readability. The index file can be created using any text editor or through the editing tools of GipGui.

2.3.2. GipGui Layout

The main screen of GipGui, shown in Figure 1, is laid out in blocks that the user will typically use in a top-to-bottom, left-to-right manner. Following is a brief description of the input blocks of the GipGui main screen:

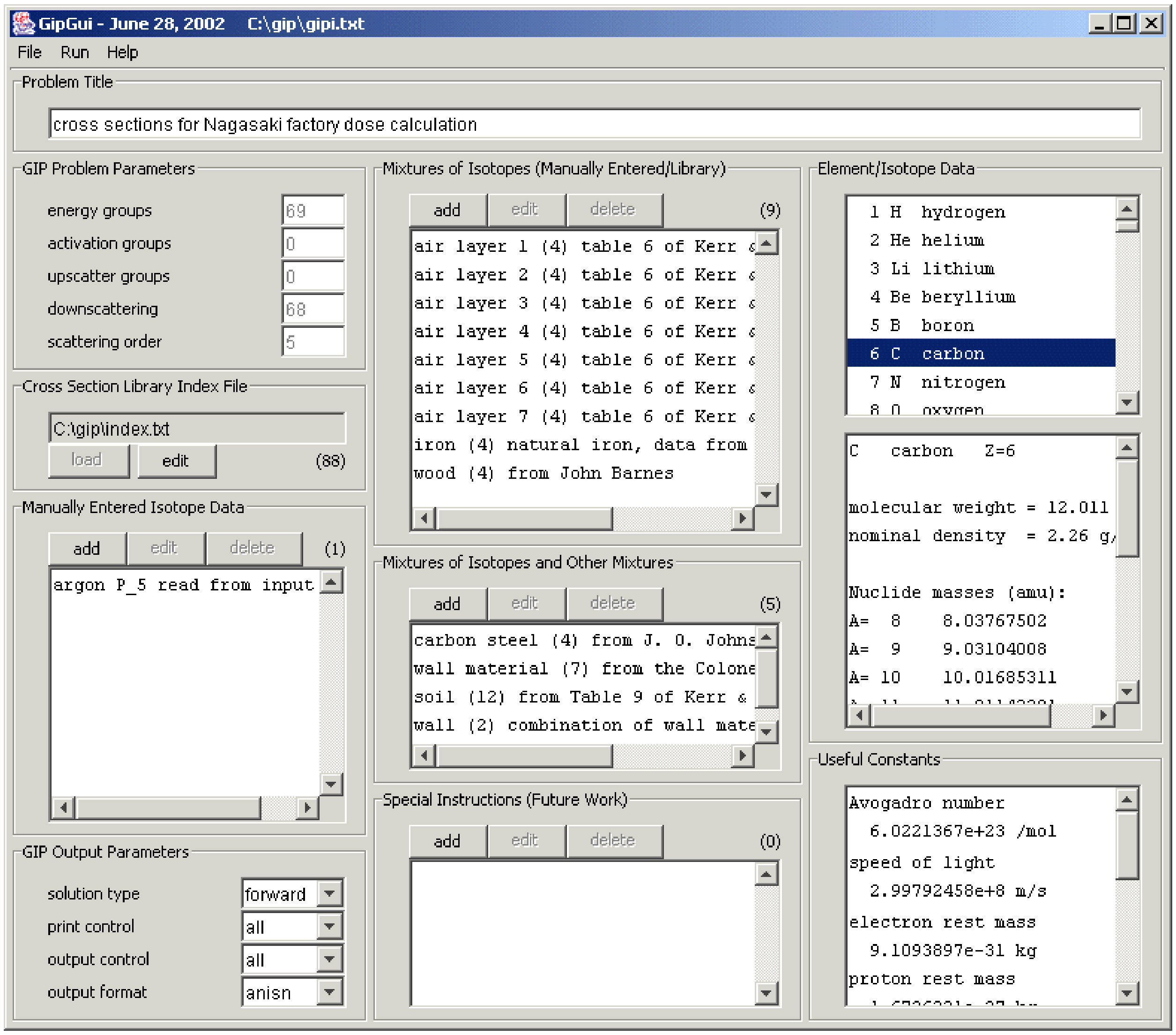
Problem Title - The problem title is a part of the standard GIP input deck.

GIP Problem Parameters - The number of energy groups, number of activation groups, number of upscatter groups and the number of downscatter groups can be specified by the user. If a library index file is being used, these four values for the GIP input deck will be forced to match those listed in the library index file. Once an index file has been loaded or isotopes have been manually entered, these numbers are no longer editable. The scattering order is editable until an isotope data is manually entered.

Cross-Section Library Index File - If one is not specified at the start of GipGui, the user may load or create a new index file. Once one is loaded, the user may edit it. Note that this will not change the library, since the library index file is independent of the actual library.

Manually Entered Isotope Data - This allows the user to add, edit and delete the isotopes entered into GIP manually (not through a library) as is often done in simple testing configurations. The list shows the currently defined isotopes. To edit or delete an isotope, click on it and then select the edit or delete button. A new isotope can be added at any time.

GIP Output Parameters - Standard part of GIP input, these describe what you want GIP to produce.



```
Problem Title
cross sections for Nagasaki factory dose calculation

GIP Problem Parameters
energy groups          69
activation groups      0
upscatter groups       0
downscattering         68
scattering order       5

Cross Section Library Index File
C:\gip\index.txt                              (88)

Manually Entered Isotope Data                 (1)
argon P_5 read from input

GIP Output Parameters
solution type          forward
print control          all
output control         all
output format          anisn

Mixtures of Isotopes (Manually Entered/Library)   (9)
air layer 1 (4) table 6 of Kerr &
air layer 2 (4) table 6 of Kerr &
air layer 3 (4) table 6 of Kerr &
air layer 4 (4) table 6 of Kerr &
air layer 5 (4) table 6 of Kerr &
air layer 6 (4) table 6 of Kerr &
air layer 7 (4) table 6 of Kerr &
iron (4) natural iron, data from
wood (4) from John Barnes

Mixtures of Isotopes and Other Mixtures           (5)
carbon steel (4) from J. O. Johns
wall material (7) from the Colone
soil (12) from Table 9 of Kerr &
wall (2) combination of wall mate

Special Instructions (Future Work)                (0)

Element/Isotope Data
 1 H  hydrogen
 2 He helium
 3 Li lithium
 4 Be beryllium
 5 B  boron
 6 C  carbon
 7 N  nitrogen
 8 O  oxygen

C    carbon   Z=6

molecular weight = 12.011
nominal density  = 2.26 g/

Nuclide masses (amu):
A=  8    8.03767502
A=  9    9.03104008
A= 10    10.01685311

Useful Constants
Avogadro number
  6.0221367e+23 /mol
speed of light
  2.99792458e+8 m/s
electron rest mass
  9.1093897e-31 kg
proton rest mass
```

Figure 1. The main screen for GipGui.

Mixtures of Isotopes (Manually Entered/Library) - This is the heart of GIP, the specification of mixtures of library materials and the manually entered isotope data. Typically, the library and manually entered isotope data are microscopic cross sections (in units of barns) and they are usually mixed to generate macroscopic cross sections (in units of per cm). Macroscopic cross sections can also be combined to form other macroscopic cross sections.

The list shows the currently defined mixtures. To edit or delete a mixture, click on it and then select the edit or delete button. A mixture can be added at any time. When adding a new mixture, there are several different ways to specify the combination of the individual ingredients. These ways are:

— by atomic densities - in atoms/(b cm)
— by atom fractions - with total in atoms/(b cm)

— by atomic densities - in atoms/cm^3
— by atom fractions - with total in atoms/cm^3
— by mass densities - in g/cm^3
— by mass fractions - with total in g/cm^3
— by molecular formula - in atoms per molecule
— unitless - user supplies multiplier

For some of these methods, data for the mixture must be entered before the ingredients can be specified. As the user enters the required data, the factors that GIP needs for mixing materials together are computed. These are shown in a pop-up "Edit Mixture" window. Mixture ingredients may be edited in different units than they were input into the mixture, so long as any additional required data is provided by the user.

Mixtures of Isotopes and Other Mixtures - These mixtures behave just like the mixtures above, except that the list of possible ingredients is not limited to the library and isotopes, but includes all of the mixtures of isotopes and the mixtures of mixtures. These can be either microscopic or macroscopic cross sections. The real reason to split the mixtures into two groups is simply to aid the user and try to prevent unit mismatches.

As the user enters new materials (mixtures, isotopes, library materials) or changes the names of old materials, GipGui will try to provide the mass data if it recognizes the new name. The name is searched for element names, symbols and mass numbers. If an appropriate isotope is found, that mass is used, if not, the elemental data is used. For example, the names "U", "uranium231", or "U235" will set the mass to be 238.029, 231.0362892 or 235.0439231 g/mol. Using a name such as "orange metal" will give a mass of 15.9994 g/mol (the first letter is "o", as in oxygen). The user may of course override these guessed masses with his own values.

Special Instructions (Future Work) - This is not functional at this time. In the future, this will hold all of the special instructions that GIP is capable of handling, such as multiplying a mixture by a constant, or doing something to one P_n component of a material, instead of all the components of that material.

Element/Isotope Data - This is provided as a service to the user. Click on any element and its isotopic data will appear in the lower scroll pane. When entering mixture ingredients, the user invariably needs this sort of data.

Useful Constants - This is also provided to the user as a simple convenience.

2.4. Restrictions or Limitations

GipGui is the initial step in developing a modern cross-section manipulation and maintenance software to replace GIP. The primary purpose of GipGui is to provide a modern interface that is compatible with existing GIP input files. As such GipGui is not designed to handle some unnecessary, and seldom used, capabilities of GIP. Following is a list of presently known limitations, and users are encouraged to communicate to the author any additional limitations they discover:

1. "=gip" and "=end" not permitted:

   Limitation: The JDOS instructions "=gip" at the beginning, and "=end" at the end, of a GIP input file cause GipGui to crash.

   Solution: Remove these lines before opening the input file with GipGui.

2. Stacked cases:

   Limitation: JDOS allows the user to stack GIP input data for multiple runs in the same input file, changing the unit number of the binary output file between runs. GipGui processes only the first case in a sequence of runs of this sort.

   Solution: Separate the input for multiple cases into separate input files prior to opening them with GipGui.

3. Only / permitted as comment character:

   Limitation: The additional comment characters ' and " cause GipGui to hang; only the comment character / is recognized by GipGui.

   Solution: Replace ' and " with /; this can be done easily via the unix line editor, ed, commands:

   ```
   1,$ s/'/\//g
   1,$ s/"/\//g
   ```

4. Empty lines read as zeros:

   Limitation: Lines with no alphanumeric characters in the middle of an array are interpreted as zeros by GipGui.

   Solution: Remove clear lines within an array input, or to preserve readability, comment out with "/".

5. 0 entry in 10$ array not permitted:

   Limitation: This is an antiquated contrivance used in one of the sample input files distributed with TORT to process kerma factors (for which there are no higher Pn moments). It is neither necessary nor intuitive, hence it is not allowed by GipGui.

   Solution: Examine the GIP input file prior to opening it with GipGui and modify the intent of zeros in the 10$ array with equivalent standard instructions.

2.5. Typical Running Time

For all cases tested, the runtime on various platforms was within the fraction of a minute time scale. Large amounts of user-input isotope data for many-group problems could take several minutes to load.

2.6. Computer Hardware Requirements

GipGui has been tested on Windows systems and Linux systems. Since it is distributed as a single JAR file, GipGui should be able to be run anywhere that Java can be installed.

2.7. Computer Software Requirements

Java 1.3 or higher is required. Java can be downloaded, free of charge, from the Sun Microsystems, Inc. web page, http://java.sun.com/

2.8. Contents of Code Package

The code itself is contained in a single JAR file. Also available are example GIP input decks and example library index files. Help is provided via an *.html file, which is embedded in the JAR file.

3. SUMMARY

GipGui is still in its very earliest development. Users of GIP are encouraged to contact the authors for a copy of GipGui to test for their own use. Feedback is greatly appreciated – the whole reason for the development of GipGui is to help the discrete ordinates community. With support and input from the users, GipGui will be improved.

ACKNOWLEDGMENTS

Oak Ridge National Laboratory is managed and operated by UT-Battelle, LLC for the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

REFERENCES

1. J. O. Johnson, Ed., “A User's Manual for MASH 1.0 - A Monte Carlo Adjoint Shielding Code System”, *ORNL/TM-11778*, Oak Ridge National Laboratory, (March 1992).
2. W. A. Rhoades and R. L. Childs, RSICC Computer Code Collection CCC-650, "DOORS-3.2, One-, Two- and Three-Dimensional Discrete Ordinates Neutron/Photon Transport Code System" (July 1998).
3. W. A. Rhoades and D. B. Simpson, “The TORT Three-Dimensional Discrete Ordinates Neutron/Photon Transport Code” *ORNL/TM-13221* (1997).
4. Y. Y. Azmy, “The Three-Dimensional, Discrete Ordinates Neutral Particle Transport Code TORT: An Overview,” *OECD/NEA Meeting on 3D Deterministic Radiation Transport Computer Programs, Features, Applications, and Perspectives*, December 2-3, 1996, Paris, France, p. 17 (1997).